11791 HW4
Jung In Lee
AndrewId: junginl

Features of the System:

I mainly modified the DocumentVectorAnnotator.java file and the RetrivalEvaluator.java file.

CollectionReader: DocumentReader.java
Annatator: DocumentVectorAnnotator.java
CasConsumer: RetrivalEvaluator.java

The given CollectionReader file basically reads in the data file. I didn't modify this much.

DocumentVectorAnnotator:

1. Constructs a vector of tokens from the input sentence.
2. Computes the Term Frequency. In order to do so, I first constructed a unique set of tokens, and then computed how many times each unique token appears in the given sentence.
3. Updates the tokenList in CAS.

RetrivalEvaluator:

1. Define necessary functions:
   a. The "dot" function: Computes the dot product between two vectors.
   b. The "computeCosineSimilarity" function: Computes the cosine similarity between two vectors, namely the question sentence and each corresponding answer candidate sentence. This function calls the "dot" function.
   c. The "compute_mrr" function: Computes the Mean Reciprocal Rank.

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

2. Compute the cosine similarity between a question sentence and each answer candidate sentence, using the "computeCosineSimilarity" function defined above.
   Note: The code would have looked much more concise, if I could have made better use of the information already processed. However, I'm still waiting for the TA to give me feedback on HW2, so that I can learn how the UIMA and its data structure works and how to implement them. For this task, I had to compute everything from scratch again. I can update the codes later (after I hear from Zi), if necessary.
3. Compute the rank of the sentence marked as "rel=1" for each question Id.
4. Then compute the MRR over all the sentences.

Bonus:
Jaccard Coefficient:

$$s = \frac{|Q \cap D|}{|Q \cup D|} = \frac{p}{p+q+d}$$

where p = # of variables that are positive for both,
q = # of variables that are positive in Q but not D
d = # of variables that are positive in D but not Q.[1]
Thus, in our case, for the two vectors queryV and docV,
p = length of the vector intersectV, which is the set intersection of queryV and docV
q = length of the queryV – length of the intersectV
d = length of the docV – length of the intersectV.

Result:

```
Score:0.2  rank=1     rel=1 qid=1 sent2
Score:0.05263157894736842  rank=1     rel=1 qid=2 sent3
Score:0.3333333333333333   rank=1     rel=1 qid=3 sent1
Score:0.05 rank=3     rel=1 qid=4 sent2
Score:0.0  rank=2     rel=1 qid=5 sent3
 (MRR) Mean Reciprocal Rank ::0.7666666666666667
Total time taken: 3.221
```

Dice Coefficient:

$$s = \frac{2|Q \cap D|}{|Q| + |D|} = \frac{p}{2p+q+d}^{2}$$

Dice coefficient is similar to Jaccard coefficient, but it gives twice more weight to the intersection of the two vectors.

Result:

```
Score:0.3333333333333333   rank=1     rel=1 qid=1 sent2
Score:0.1  rank=1     rel=1 qid=2 sent3
Score:0.5  rank=1     rel=1 qid=3 sent1
Score:0.09523809523809523  rank=3     rel=1 qid=4 sent2
Score:0.0  rank=2     rel=1 qid=5 sent3
 (MRR) Mean Reciprocal Rank ::0.7666666666666667
Total time taken: 1.423
```

[1] http://www.stanford.edu/~maureenh/quals/html/ml/node68.html
[2] http://www.stanford.edu/~maureenh/quals/html/ml/node69.html

Comparison:

For this particular set of data, the results using three different scoring metrics seem to be similar. I haven't actually calculated the p-value, but it is likely that the differences are not statistically significant. Nevertheless, the scores represent some characteristics for the metrics. Dice coefficients give more weights to the intersection, which means that in our case, if the two sentences have more number of common words, the score would be higher, and this leverage is higher than the Jaccard coefficients. This can be verified by the results, where the scores for Dice are generally higher than those for Jaccard. It is interesting to note that the scores for Cosine Similarity are all higher than those for Dice. This may be due to the denominator. For Cosine, we are dividing by the multiplication of the length of vector 1 and that of vector 2, while for Dice, we are dividing be the addition of those two values. However, if we look at the ranks, all five ranks are the same across the three different metrics, so it is difficult to conclude whether they may really generate in different results.

Error Analysis

First Attempt:

```
Score:0.45226701686664544   rank=1     rel=1 qid=1 sent2
Score:0.10206207261596574   rank=1     rel=1 qid=2 sent3
Score:0.5070925528371099    rank=1     rel=1 qid=3 sent1
Score:0.17213259316477406   rank=3     rel=1 qid=4 sent2
Score:0.0  rank=2     rel=1 qid=5 sent3
 (MRR) Mean Reciprocal Rank ::0.7666666666666667
```

Analysis:

The score for the fifth question-answers pair is 0, which means cosine similarity is 0. This indicates that in the answer sentence, there is not even one word that matches the tokens in the question sentence. I took a look at the data to figure out what the problem was.

```
qid=5 rel=99      It takes a long time to grow an old friend
qid=5 rel=1       Old friends are best
```

These two sentences definitely have some words in common, e.g., "old" and "friends." However, they are not in the exact same form. In the question sentence, the "o" in "old" is in lower case letter, while the "O" in "Old" in the answer sentence is in the upper case. Similarly, the word "friend" in the question sentence is singular, whereas the word in the answer sentence is plural.

Solution 1:

The lemmatization converts every word into its stem form. It also takes care of the lower case/ upper case differences. Thus, if I lemmatize the tokens, the code would match {"old" and "Old"} and {"friend" and "friends"}.

New result:

```
Score:0.6030226891555273    rank=1     rel=1 qid=1 sent2
Score:0.19611613513818402   rank=1     rel=1 qid=2 sent3
Score:0.4629100498862757    rank=3     rel=1 qid=3 sent1
Score:0.3333333333333333    rank=2     rel=1 qid=4 sent2
Score:0.2886751345948129    rank=1     rel=1 qid=5 sent3
 (MRR) Mean Reciprocal Rank ::0.7666666666666667
```

The score for the qid=5 definitely improved, which was expected. Everything improved except for the third problem set. I'm not sure if this is statistically significant, but I propose another solution.

Note: I kept getting errors running Stanford NLP, and I couldn't figure out what the problem was. So I instead, ran the code on the hw0.

Approach 2:

When looking at the data, we can see that in the problem set #3, we have words like "friends" and "friend." Before lemmatizing the data, the actual answer could have gotten the rank to be 1, because the word "friend" and "best" got matched up. Although the third sentence also contains the words "friend" and "best," because it has more words in the sentence that do not match with the words in the question sentence (more noise), it gets a lower cosine similarity value. In the fourth sentence, however, before the lemmatization, the word "friends" would not be matched with the word "friend" in the question sentence. With the lemmatization, however, the fourth sentence gets the greatest match (i.e. the lowest cosine similarity value), because now "friends" get matched to "friend," and it has fewer words than the actual answer. The third sentence got boosted as well, because now words like "one's" in the question sentence gets aligned with "one."

I do not think this slight degradation is due to lemma, but rather due to just the way the system is designed. Thus, I decided to keep the lemma feature, and propose another method, which is eliminating the punctuations.

Result:

```
Score:0.6030226891555273    rank=1     rel=1 qid=1 sent2
Score:0.20412414523193148   rank=1     rel=1 qid=2 sent3
Score:0.5070925528371099    rank=3     rel=1 qid=3 sent1
Score:0.3442651863295481    rank=2     rel=1 qid=4 sent2
Score:0.2886751345948129    rank=1     rel=1 qid=5 sent3
 (MRR) Mean Reciprocal Rank ::0.7666666666666667
```

Removing the punctuations has improved some of the scores. Yet, whether this is statistically significant is uncertain. We need more data to determine this.